Computer Modeling in Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2023.043247

ARTICLE

Distributed Stochastic Optimization with Compression for Non-Strongly Convex Objectives

Xuanjie Li and Yuedong Xu*

School of Information Science and Technology, Fudan University, Shanghai, China

*Corresponding Author: Yuedong Xu. Email: ydxu@fudan.edu.cn

Received: 26 June 2023 Accepted: 20 October 2023 Published: 30 December 2023

ABSTRACT

We are investigating the distributed optimization problem, where a network of nodes works together to minimize a global objective that is a finite sum of their stored local functions. Since nodes exchange optimization parameters through the wireless network, large-scale training models can create communication bottlenecks, resulting in slower training times. To address this issue, CHOCO-SGD was proposed, which allows compressing information with arbitrary precision without reducing the convergence rate for strongly convex objective functions. Nevertheless, most convex functions are not strongly convex (such as logistic regression or Lasso), which raises the question of whether this algorithm can be applied to non-strongly convex functions. In this paper, we provide the first theoretical analysis of the convergence rate of CHOCO-SGD on non-strongly convex objectives. We derive a sufficient condition, which limits the fidelity of compression, to guarantee convergence. Moreover, our analysis demonstrates that within the fidelity threshold, this algorithm can significantly reduce transmission burden while maintaining the same convergence rate order as its no-compression equivalent. Numerical experiments further validate the theoretical findings by demonstrating that CHOCO-SGD improves communication efficiency and keeps the same convergence rate order simultaneously. And experiments also show that the algorithm fails to converge with low compression fidelity and in time-varying topologies. Overall, our study offers valuable insights into the potential applicability of CHOCO-SGD for non-strongly convex objectives. Additionally, we provide practical guidelines for researchers seeking to utilize this algorithm in real-world scenarios.

KEYWORDS

Distributed stochastic optimization; arbitrary compression fidelity; non-strongly convex objective function

1 Introduction

In modern machine learning problems, datasets are frequently too large to be processed by a single machine or may be distributed across multiple computing nodes due to legal restrictions or cost considerations. In such scenarios, distributed learning has emerged as a viable solution to train models in a decentralized fashion. Each computing node trains on its local data samples, and communicates with other nodes to exchange information and update model parameters. This approach can significantly reduce processing time and has gained increasing attention in recent years.

Formally, we consider an optimization problem of the form

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathbb{R}^d} \left[f(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x}) \right], \tag{1}$$

where $f_i : \mathbb{R}^d \to \mathbb{R}$ is the local objective function of computing node (alternatively referred to as a worker) i for $i \in \{1, \cdots, n\}$, $\mathbf{x}$ usually refers to the neural network model parameter, and each f_i is determined by the local dataset of node i, i.e.,

$$f_i(\mathbf{x}) = \frac{1}{m_i}\sum_{j=1}^{m_i} F_i(\mathbf{x}, \zeta_j), \tag{2}$$

where m_i denotes the number of data samples of node i, ζ_j represents one data sample, and $F_i : \mathbb{R}^d \times \Omega \to \mathbb{R}$ is the loss function. For example, in a regression problem, a square loss function is commonly used as $F_i(\cdot)$.

Consensus-based gradient descent algorithms are extensively researched and employed in distributed learning owing to their simplicity and efficiency. These algorithms enable nodes to exchange parameters with neighboring nodes and incorporate new information from received messages to drive the parameters toward the global average parameter value. Meanwhile, each node computes gradients using its local data samples and updates its parameters using the gradient descent methods.

One limitation of these algorithms is the requirement for exact parameter transmission among nodes, which can be challenging in scenarios where there are restrictions on data transmission amount or high communication latency (e.g., with large neural networks [1] or limited network bandwidth). In such cases, compressing the parameters becomes necessary. However, parameter compression usually results in a decrease in the convergence rate. To address this challenge, Koloskova et al. proposed CHOCO-SGD [2], which was a compressed distributed algorithm designed for strongly convex objective functions. CHOCO-SGD allows for both biased and unbiased compression operators and can achieve arbitrary compression precision. Arbitrary compression precision means that any compression fidelity $0 < \omega \leq 1$ ($w = 1$ indicating no compression) can guarantee the same convergence rate order as the no-compression equivalent.

However, strong convexity is a restrictive assumption that may not hold in various practical applications, such as in operations research and machine learning. For instance, consider the Lasso problem [3] with an objective function represented as $\sum_{i=1}^{m} \frac{1}{2}(\langle d_i, \mathbf{x}\rangle - l_i)^2 + \beta\|\mathbf{x}\|_1$, where $i \in \{1, \cdots, m\}$ denotes the data index, β denotes the regularization coefficient, and d_i and l_i represent data features and labels, respectively. In this case, each component function is non-strongly convex, as well as the global objective. Another well-known example is logistic regression [4], where the objective is $\sum_{i=1}^{m} \log(1 + \exp(-l_i\langle d_i, \mathbf{x}\rangle))$. Therefore, it is crucial to investigate the behavior and performance of CHOCO-SGD for objectives that are non-strongly convex in nature. The main contributions are summarized as follows:

- We establish a compression fidelity bound for CHOCO-SGD, ensuring its convergence over non-strongly convex functions. This criterion serves as a valuable guideline for researchers and engineers when applying CHOCO-SGD in practical scenarios.
- We rigorously prove that within the compression fidelity threshold, CHOCO-SGD achieves the same convergence rate order as DSGD over non-strongly convex objectives, and compression

fidelity only affects the higher-order terms in the rate. This means that CHOCO-SGD effectively reduces the transmitted data without compromising training efficiency.

- We conduct comprehensive experiments to validate our theoretical results. The numerical outcomes show that when controlling compression fidelity above the lower bound, CHOCO-SGD effectively reduces communication costs, while remains a comparable convergence rate to that of DSGD. Additionally, our experiments reveal that CHOCO-SGD fails to converge when the fidelity falls below acceptable limits, implying that this algorithm cannot achieve arbitrary compression fidelity on non-strongly convex objectives. Moreover, we also observe that CHOCO-SGD cannot converge in time-varying network topologies.

Notations. In this paper, we use uppercase letters like X, A to denote various matrices, including parameter matrix, gradient matrix, weight matrix, etc. Lowercase letters with subscripts are used to denote vectors, for example, $\mathbf{x}_i$ means the parameter vector of node i. In terms of some special matrices and vectors, we use $\mathbf{1}$ for a column vector whose elements are all 1 and $\mathbf{I}$ for the identity matrix whose size can be inferred from the context. We uniformly use $\|\cdot\|$ without subscripts for norm notations. When the norm is on a vector, it should represent the ℓ_2 norm, but when it is on a matrix, it should be the Frobenius norm.

2 Related Work

Consensus algorithm. For the distributed optimization problem, we want every node to reach an agreement on the global parameter value. The agreement means finding the average of n initial values stored on n nodes [5,6]. So, the consensus problem can be regarded as a subproblem of distributed optimization problem. There are a plethora of research works conducted to explore the consensus problem. It has been proved that the consensus problem can be addressed in the sense of linear convergence on undirected graph [6].

Distributed optimization. On the basis of consensus algorithms, researchers proposed many distributed gradient-based algorithms [7–12]. In undirected topology, Nedic et al. [7] and Yuan et al. [8] analyzed the convergence of Decentralized Stochastic Gradient Descent algorithm (DSGD) on non-strongly convex objective functions. Lian et al. proved that for nonconvex and Lipschitz-smooth objective functions, DSGD can converge to a stationary point with the rate $O(\frac{1}{\sqrt{nT}})$ [13], where T denotes the number of episodes. Furthermore, Shi et al. proposed the EXTRA algorithm that had linear convergence rate for strongly convex objectives [14]. Assran et al. further addressed challenges introduced by directed graphs by designing the SGP algorithm based on DSGD [15]. More related algorithms has been reviewed in [16,17].

Parameter/gradient compression. With the rapid growth of learning models, traditional distributed learning faces significant communication pressure. As a result, Researchers have increasingly focused on designing communication-efficient algorithms [2,18–24]. The proposed methods can generally be classified into three main categories. The first category is quantization. Instead of transmitting precise vectors (such as model parameters or gradients), nodes exchange limited bits that approximate the original vectors. The second category is sparsification, where only a few elements of a vector are transmitted accurately, while the remaining elements are forced to be zero. The third category involves transmitting parameters in some iterations rather than in every iteration. This aims to reduce the amount of transmitted information. All these methods aim to find the optimal parameter value by transmitting less information. Therefore, in this paper, we use the term *compression* to encompass

these methods. Kolovaskova et al. proposed the CHOCO-SGD algorithm [2], a distributed stochastic algorithm over undirected graphs. This algorithm allows for arbitrary compression precision and has been proven to converge for strongly convex objectives. Taheri et al. [25] extended this algorithm to directed graphs for both convex and nonconvex objectives based on the SGP algorithm [15]. Different from this work, our study is based on undirected graphs. Table 1 further illustrates the highlights of this work and the differences between the work involved.

Table 1: Comparisons on different algorithms for decentralized optimization problem

Literature	Algorithm	Applied technology	Graph	Condition	Rate
[15]	SGP	DSGD + Push-sum	Directed graph	Non-convex	$O(\frac{1}{\sqrt{nT}})$
[25]	Quantized push-sum	Compressed + SGP	Directed graph	Non-convex	$O(\frac{1}{\sqrt{nT}})$
This work	CHOCO-SGD	Compressed + DSGD	Undirected graph	Non-strongly convex	$O(\frac{1}{\sqrt{nT}})$

3 CHOCO-SGD Algorithm

In this section, we will introduce the basic knowledge of distributed optimization and present the classic Distributed Stochastic Gradient Descent (DSGD) algorithm. Additionally, we will discuss the fundamental ideas underlying the CHOCO-SGD algorithm.

We assume that the computing nodes are distributed across an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ represent the set of nodes and edges, respectively. The DSGD algorithm is an efficient method to solve the distributed optimization problem in Eq. (1). It consists of two key components: *consensus*, also referred to as *gossip*, and parameter updates. The consensus step aims to achieve parameter agreement among all nodes. Specifically, we denote the local parameter of node i as $\mathbf{x}_i$, and perform the following parameter aggregation process:

$$\mathbf{x}_i(t+1) = \sum_{j=1}^{n} a_{ij}\mathbf{x}_j(t), \tag{3}$$

where t denotes the iteration number and a_{ij} denotes the weight assigned by node i to node j. For compatibility with the underlying graph structure, a_{ij} is set to be positive if and only if edge$(i,j) \in \mathcal{E}$ or $i = j$. This ensures that only neighboring nodes, as determined by the graph, can communicate and exchange their parameters. By using a matrix $A = (a_{ij})_{n\times n}$ [1] to incorporate all the weights, it has been proven that when A is row stochastic (i.e., each row of A sums to one and all the entries are non-negative), the parameters of all nodes can reach a consensus. This implies that each parameter $\mathbf{x}_i(t)$ will converge to the average of the initial values $\frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i(1)$ [26]. This consensus algorithm enables the nodes to synchronize their parameter values.

[1] a_{ij} represents the entry in the i^{th} row and the j^{th} column of A.

The second component of the DSGD algorithm involves parameter updates using stochastic gradient descent. Combining these components, the update rule of DSGD is given as follows:

$$\mathbf{x}_i(t+1) = \sum_{j=1}^{n} a_{ij}\mathbf{x}_j(t) - \alpha \nabla F_i(\mathbf{x}_i(t), \zeta_{i,t}), \tag{4}$$

where $\nabla F_i(\mathbf{x}_i(t), \zeta_{i,t})$ represents the gradient of the local objective function evaluated at $\mathbf{x}_i(t)$ using the data batch $\zeta_{i,t}$, and α denotes the learning rate. This update rule combines the consensus step, which ensures parameter agreement, with the gradient descent step, which drives the parameters towards optimal values. We formally present the DSGD algorithm in Algorithm 1. Under some constraints, extensive studies [16] have demonstrated that the DSGD algorithm achieves a convergence rate of $O(\frac{1}{\sqrt{nT}})$ on non-strongly convex objective functions, where T denotes the number of episodes.

Algorithm 1: DSGD

Input: Initial parameters $\mathbf{x}_i(1) = 0$ for $i \in \mathcal{N}$, step size α, communication graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ and weight matrix A

1: **for** each node $i \in \{1, \cdots, n\}$ and $t \in \{1, \cdots, T\}$ **do**
2: **for** neighbors j such that $(i, j) \in \mathcal{E}$ **do**
3: send $\mathbf{x}_i(t)$ to j and receive $\mathbf{x}_j(t)$ from j
4: **end for**
5: temp $= \sum_{j=1}^{n} a_{ij}(\mathbf{x}_j(t) - \mathbf{x}_i(t))$
6: $\mathbf{z}_i(t+1) = \mathbf{x}_i(t) +$ temp
7: $\mathbf{x}_i(t+1) = \mathbf{z}_i(t+1) - \alpha \nabla F_i(\mathbf{x}_i(t), \zeta_{i,t})$
8: **end for**

However, when dealing with massive training models, the DSGD algorithm encounters a communication bottleneck. To address this limitation, a straightforward yet effective solution is to compress the parameters before transmitting them over the network. For this purpose, We introduce a compression operator denoted as Q, along with the definition of compression fidelity ω, which indicates the ratio of preserved information. A higher value of ω signifies a greater fidelity, implying that less information is discarded during the compression process. The value of ω satisfies the following assumption.

Assumption 3.1. The compression operator Q satisfies

$$\mathbb{E}_Q \|Q(\mathbf{x}) - \mathbf{x}\|^2 \leq (1-\omega)\|\mathbf{x}\|^2.$$

We take the expectation in this context because there might be inherent randomness in the compression operators, such as in the case of randomized gossip discussed later. It is worth noting that when $\omega = 1$, the compressed parameter $Q(\mathbf{x})$ should ideally be the same as $\mathbf{x}$ (almost surely, although we may disregard negligible difference). And we state that the operator Q can be either biased or unbiased, encompassing a wide range of compression techniques.

Example 3.1. Examples of compression

- randomized gossip:

$$Q(\mathbf{x}) = \begin{cases} \mathbf{x}, \text{with probability } p \\ \mathbf{0}, \text{with probability } 1-p \end{cases},$$

where $p \in (0, 1]$. We can find $\omega = p$.

- Top-k sparsification [20]: for a parameter vector $\mathbf{x} \in \mathbb{R}^d$, we select k dimensions with the highest magnitude, where $k \in \{1, 2, \cdots, d\}$. Subsequently, we transmit a compressed version of the parameter vector, retaining only the selected k items while setting all other positions to 0. It is worth noting that the compression fidelity in this case can be calculated as $\omega = \frac{k}{d}$.

Through conducting simple experiments, one can observe that when integrating compression directly into the transmitted parameter in Algorithm 1 (i.e., replacing the transmission of $\mathbf{x}_i(t)$ with $Q(\mathbf{x}_i(t))$), the algorithm typically fails to converge. The primary reason behind this behavior stems from the magnitude of $\mathbf{x}_i(t)$. In general, it lacks a bound, meaning its magnitude is not necessarily close to zero. As a result, the error introduced by compressing (such as forcing certain positions to be zero) becomes difficult to control. Hence, it is important to employ the compression operator appropriately and judiciously.

Based on Assumption 3.1, CHOCO-SGD was proposed and is presented in Algorithm 2. As the transmitted parameters are compressed and inexact, each node i needs to maintain n additional auxiliary variables $\{\hat{\mathbf{x}}_j(t), j \in \mathcal{N}\}$ to approximate the true parameters of all n nodes. Specifically, for each node $i \in \mathcal{N}$, the newly added auxiliary variable $\hat{\mathbf{x}}_j(t)$ represents an estimation of the actual local parameter of node j. In Algorithm 2, instead of sending the compressed parameters, the nodes transmit the compressed value of the difference between the true local parameter and its estimator. Subsequently, upon receiving messages from neighboring nodes, every node updates its estimators of parameters of other nodes, denoted as $\{\hat{\mathbf{x}}_1(t+1), \cdots, \hat{\mathbf{x}}_n(t+1)\}$. If the estimations maintained by each node are accurate, the error introduced by compression is minimal and will eventually vanish. It has been proven that $Q(\hat{\mathbf{x}}_i(t) - \mathbf{x}_i(t))$ asymptotically equals to $\hat{\mathbf{x}}_i(t) - \mathbf{x}_i(t)$, which does not hold for $Q(\mathbf{x}_i(t))$ and $\mathbf{x}_i(t)$.

Algorithm 2: CHOCO-SGD

Input: Initiate parameters $\mathbf{x}_i(1) = 0$ for $i \in \mathcal{N}$, SGD stepsize α, communication graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, weight matrix A and all auxiliary variables $\hat{x}_i(1) = 0, i \in \mathcal{N}$

for each node $i \in \{1, \cdots, n\}$ and $t \in \{1, \cdots, T\}$ **do**
  $Q_i(t) = Q(\mathbf{x}_i(t) - \hat{\mathbf{x}}_i(t))$
  for neighbors j such that $(i, j) \in \mathbf{E}$ **do**
    send $Q_i(t)$ to j and receive $Q_j(t)$ from j
    $\hat{\mathbf{x}}_j(t+1) = \hat{\mathbf{x}}_j(t) + Q_j(t)$
  end for
  temp $= \sum_{j=1}^{n} a_{ij}(\hat{\mathbf{x}}_j(t+1) - \hat{\mathbf{x}}_i(t+1))$
  $\mathbf{z}_i(t+1) = \mathbf{x}_i(t) + \text{temp}$
  $\mathbf{x}_i(t+1) = \mathbf{z}_i(t+1) - \alpha \nabla F_i(\mathbf{x}_i(t), \zeta_{i,t})$
end for

Koloskova et al. [2] has proved that Algorithm 2 achieved a convergence rate of $O(\frac{1}{T})$ on strongly convex objective functions when specific parameters such as the step size and regularization parameter were appropriately chosen. Furthermore, it has been established that this algorithm can converge with arbitrary compression fidelity, indicating minimal communication pressure. To evaluate its potential in practical scenarios, we investigate its convergence properties on non-strongly convex objective functions.

4 Convergence Analysis for Non-Strongly Convex Objectives

In this section, we begin by introducing assumptions regarding the underlying graph structure. As mentioned previously, we use $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to denote the undirected graph and use $\mathcal{N}_i$ to denote all the neighbors of node i together with itself, i.e., $\mathcal{N}_i = \{j : (i,j) \in \mathcal{E}\} \cup \{i\}$. We have a couple of well-accepted assumptions in distributed optimization, which are related to the graph structure and the corresponding weight matrix.

Assumption 4.1. The underlying graph structure $\mathcal{G}$ is connected.

Assumption 4.2. Given the aggregation weights a_{ij}, the weight matrix $A = (a_{ij})_{n\times n}$ is doubly stochastic and non-negative, i.e.,

$$A \cdot \mathbf{1} = \mathbf{1}, \quad A^{\mathrm{T}}\mathbf{1} = \mathbf{1}, \quad a_{ij} \geq 0,$$

where A satisfies $\sigma_2 < 1$, where σ_2 denotes the second largest singular value of A.

Remark 4.1. Under the undirected graph setting, a doubly stochastic matrix can be easily found. Given an undirected graph $\mathcal{G}$, with its adjacent matrix A_0, the degree matrix D and the maximum degree Δ, it can be directly shown that $\dfrac{A_0 + (\Delta+1)\mathbf{I} - D}{\Delta + 1}$ is a valid doubly stochastic matrix [27].

We next introduce several basic assumptions on the objective function and its gradients. Note that these assumptions are common in first-order continuous optimization.

Assumption 4.3. Each local objective function $f_i(\cdot)$ is convex, i.e.,

$$f_i(\mathbf{y}) \geq f_i(\mathbf{x}) + \langle \nabla f_i(\mathbf{x}), \mathbf{y} - \mathbf{x}\rangle, \quad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d,$$

and L-smooth, i.e., there exists a constant $L > 0$ (for all $i \in \mathcal{N}$), such that

$$f_i(\mathbf{y}) \leq f_i(\mathbf{x}) + \langle \nabla f_i(\mathbf{x}), \mathbf{y} - \mathbf{x}\rangle + \frac{L}{2}\|\mathbf{y} - \mathbf{x}\|^2, \quad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d.$$

The definition of L-smooth is equivalent to the following inequality:

$$\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\| \leq L\|x - y\|, \quad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d.$$

Assumption 4.4. There exists a constant M, such that

$$\mathbb{E}_{\zeta_{i,t}}\|\nabla F_i(\mathbf{x})\|^2 \leq M^2, \quad \forall \mathbf{x} \in \mathbb{R}^d.$$

Assumption 4.5. There exists a constant σ, such that

$$\mathbb{E}_{\zeta_{i,t}}\|\nabla F_i(\mathbf{x}) - \nabla f_i(\mathbf{x})\|^2 \leq \sigma^2, \quad \forall \mathbf{x} \in \mathbb{R}^d.$$

Assumptions 4.4 and 4.5 bound the local gradients and variance with respect to data samples, respectively. The latter assumption can be easily satisfied when the dataset across nodes is independent and identically distributed. Besides, the following asymptotic property of A implies that all the elements in A^t will be distributed in the vicinity of $\dfrac{1}{m}$ as $t \to \infty$.

Lemma 4.1. If $A \in \mathbb{R}^{n\times n}$ is a doubly stochastic matrix with $\sigma_2 < 1$, then there exists constants $\lambda \in (0,1)$ and $C > 0$, such that

$$\|A^t - \frac{1}{n}\mathbf{1}\mathbf{1}^T\| \leq C\lambda^t,$$

where A^t refers to the product of t matrices.

This lemma states that as matrix multiplication progresses, all elements of A^t converge to $\frac{1}{n}$. The detailed proof is presented in the appendix. With these assumptions and lemma, we can analyze the convergence properties of CHOCO-SGD for non-strongly convex objectives.

Theorem 4.1. Under Assumptions 3.1 and 4.1–4.5, and assume $\alpha \leq \frac{1}{8L}$, the CHOCO-SGD algorithm satisfies

$$\frac{1}{T}\sum_{t=1}^{T}\left(\mathbb{E}f(\bar{\mathbf{x}}(t)) - f(\mathbf{x}^*)\right) \leq \frac{\mathbb{E}\|\mathbf{x}^*\|^2}{T\alpha} + \frac{2\sigma^2\alpha}{n} + \frac{24C^2\beta L\alpha^2}{(1-\lambda)^2} + \frac{6C^2M^2L\alpha^2}{(1-\lambda)^2}, \tag{5}$$

where

$$\omega > 1 - \frac{1}{3 + 24n + \frac{384n^2C^2}{(1-\lambda)^2}}, \tag{6}$$

and $\bar{\mathbf{x}}(t)$ denotes the average parameter after the t iterations, i.e., $\bar{\mathbf{x}}(t) = \frac{X(t)\mathbf{1}}{n}$. Notations C and λ come from Lemma 4.1, and

$$\beta = \frac{3(1-\omega)nM^2 + \frac{96(1-\omega)C^2n^2M^2}{(1-\lambda)^2}}{1 - \left[(3+24n)(1-\omega) + \frac{384n^2(1-\omega)C^2}{(1-\lambda)^2}\right]}. \tag{7}$$

Let $\alpha = O\left(\frac{n}{T}\right)$, the right-hand side (RHS) of Eq. (5) will be equal to $O\left(\frac{1}{\sqrt{nT}}\right) + O\left(\frac{n}{T}\right)$. When T is large enough, the term $\frac{n}{T}$ is dominated by another term $\frac{1}{\sqrt{nT}}$ and the rate becomes $O\left(\frac{1}{\sqrt{nT}}\right)$, which means that the convergence rate will be the same as DSGD (Algorithm 1) on non-strongly convex objectives, i.e., $O\left(\frac{1}{\sqrt{nT}}\right)$. We can also discover that the compression fidelity ω and the graph spectral gap λ only appear in higher order terms $O\left(\frac{n}{T}\right)$ and have negligible influence on the convergence speed.

Furthermore, the expression $\omega > 1 - \frac{1}{3 + 24n + \frac{384n^2C^2}{(1-\lambda)^2}}$ is a sufficient condition of achieving $O\left(\frac{1}{\sqrt{nT}}\right)$ convergence rate. In this constraint, ω needs to be higher than a threshold related to the number of nodes and communication topology. We will demonstrate with experiments that CHOCO-SGD cannot converge on non-strongly convex objectives with low compression fidelity. In other words, the CHOCO-SGD algorithm does not allow arbitrary compression fidelity.

Proof. For simplicity, we let

$$X(t) = (\mathbf{x}_1(t), \cdots, \mathbf{x}_n(t))$$

$$\hat{X}(t) = (\hat{\mathbf{x}}_1(t), \cdots, \hat{\mathbf{x}}_n(t))$$

$$Z(t) = (\mathbf{z}_1(t), \cdots, \mathbf{z}_n(t))$$

$$\nabla(t) = (\nabla F_1(\mathbf{x}_1(t), \zeta_{1,t}), \cdots, \nabla F_n(\mathbf{x}_n(t), \zeta_{n,t}))$$

and then the updating rule of Algorithm 2 can be expressed as

$$\begin{cases} Q(t) = Q(X(t) - \hat{X}(t)) \\ \hat{X}(t+1) = \hat{X}(t) + Q(t) \\ Z(t+1) = X(t) + \hat{X}(t+1)(A - I) \\ X(t+1) = Z(t+1) - \alpha\nabla(t) \end{cases} \tag{8}$$

Note that when we use the compression operator Q on a matrix, we mean compressing every column of that matrix. Since the underlying topology is fixed, node i and node j ($i, j \in \mathcal{N}_k$) have the same estimator $\hat{\mathbf{x}}_k$ for their common neighbor node k. Then we can use one matrix $\hat{X}(t)$ to denote the original $n \times n$ auxiliary variables. Now, we start the proof by analyzing the distance between the average parameter in the $(t+1)^{th}$ iteration, denoted as $\bar{\mathbf{x}}(t+1)$, and the optimum, denoted as $\mathbf{x}^*$. By the updating rule and properties of A, we have

$$\begin{aligned} X(t+1)\mathbf{1} &= X(t)\mathbf{1} + \hat{X}(t+1)(A - \mathbf{I})\mathbf{1} - \alpha\nabla(t)\mathbf{1} \\ &= X(t)\mathbf{1} - \alpha\nabla(t)\mathbf{1}. \end{aligned} \tag{9}$$

Dividing both sides by n and we can get $\bar{\mathbf{x}}(t+1) = \bar{\mathbf{x}}(t) - \alpha\bar{\nabla}(t)$, where $\bar{\nabla}(t)$ denotes the average gradient of $\nabla F_i(\mathbf{x}_i(t), \zeta_{i,t}), i \in \mathcal{N}$, i.e., $\bar{\nabla}(t) = \dfrac{\nabla(t)\mathbf{1}}{n}$. Then there exists

$$\mathbb{E}\|\bar{\mathbf{x}}(t+1) - \mathbf{x}^*\|^2 = \mathbb{E}\|\bar{\mathbf{x}}(t) - \mathbf{x}^*\|^2 + \alpha^2\mathbb{E}\|\bar{\nabla}(t)\|^2 - 2\alpha\mathbb{E}\langle\bar{\nabla}(t), \bar{\mathbf{x}}(t) - \mathbf{x}^*\rangle. \tag{10}$$

The second term can be processed as follows:

$$\mathbb{E}\|\bar{\nabla}(t)\|^2 = \mathbb{E}\|\frac{1}{n}\sum_{i=1}^{n}\nabla F_i(\mathbf{x}_i(t), \zeta_{i,t})\|^2 \le 2\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}(\nabla F_i(\mathbf{x}_i(t), \zeta_{i,t}) - \nabla f_i(\mathbf{x}_i(t)))\right\|^2 + 2\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i(t))\right\|^2, \tag{11}$$

where the inequality comes from Cauchy-Schwarz inequality. The two terms in the right-hand side (RHS) of the last equation can be bounded as follows:

$$\begin{aligned} &2\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}(\nabla F_i(\mathbf{x}_i(t), \zeta_{i,t}) - \nabla f_i(\mathbf{x}_i(t)))\right\|^2 = \frac{2}{n^2}\sum_{i=1}^{n}\mathbb{E}\left\|\nabla F_i(\mathbf{x}_i(t), \zeta_{i,t}) - \nabla f_i(\mathbf{x}_i(t)))\right\|^2 \\ &\quad + \frac{2}{n^2}\sum_{i\neq i'}\mathbb{E}\Big\langle\nabla F_i(\mathbf{x}_i(t), \zeta_{i,t}) - \nabla f_i(\mathbf{x}_i(t))),\ \nabla F_{i'}(\mathbf{x}_{i'}(t), \zeta_{i',t}) - \nabla f_{i'}(\mathbf{x}_{i'}(t)))\Big\rangle \\ &= \frac{2}{n^2}\sum_{i=1}^{n}\mathbb{E}\left\|\nabla F_i(\mathbf{x}_i(t), \zeta_{i,t}) - \nabla f_i(\mathbf{x}_i(t)))\right\|^2 \\ &\quad + \frac{2}{n^2}\sum_{i\neq i'}\mathbb{E}\Big\langle\nabla F_i(\mathbf{x}_i(t), \zeta_{i,t}) - \nabla f_i(\mathbf{x}_i(t))),\ \mathbb{E}_{i'}\nabla F_{i'}(\mathbf{x}_{i'}(t), \zeta_{i',t}) - \nabla f_{i'}(\mathbf{x}_{i'}(t)))\Big\rangle \end{aligned}$$

$$= \frac{2}{n^2}\sum_{i=1}^{n}\mathbb{E}\left\|\nabla F_i(\mathbf{x}_i(t),\zeta_{i,t}) - \nabla f_i(\mathbf{x}_i(t))\right\|^2 \le \frac{2\sigma^2}{n}, \tag{12}$$

where the last inequality applies Assumption 4.5. And for the second term, we have

$$2\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i(t))\right\|^2 \le \frac{4}{n}\sum_{i=1}^{n}\mathbb{E}\left\|\nabla f_i(\mathbf{x}_i(t)) - \nabla f_i(\bar{\mathbf{x}}(t))\right\|^2 + 4\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\bar{\mathbf{x}}(t))\right\|^2$$

$$\le \frac{4L^2}{n}\mathbb{E}\|X(t) - \bar{\mathbf{x}}(t)\mathbf{1}^{\mathrm{T}}\|^2 + 8L(\mathbb{E}f(\bar{\mathbf{x}}(t)) - f(\mathbf{x}^*)), \tag{13}$$

where the last inequality comes from Assumption 4.3 and the following result related to $\|\nabla f_i(\bar{\mathbf{x}}(t))\|^2$:

$$f(\mathbf{x}^*) \le f(\mathbf{x} - \frac{1}{L}\nabla f(\mathbf{x})) \le f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), -\frac{1}{L}\nabla f(\mathbf{x})\rangle + \frac{L}{2}\|\frac{1}{L}\nabla f(\mathbf{x})\|^2 = f(\mathbf{x}) - \frac{1}{2L}\|\nabla f(\mathbf{x})\|^2. \tag{14}$$

Plugging the results in Eqs. (12) and (13) back into Eq. (11) yields the bound for $\mathbb{E}\|\bar{\nabla}(t)\|^2$

$$\mathbb{E}\|\bar{\nabla}(t)\|^2 \le \frac{2\sigma^2}{n} + \frac{4L^2}{n}\mathbb{E}\|X(t) - \bar{\mathbf{x}}(t)\mathbf{1}^{\mathrm{T}}\|^2 + 8L(\mathbb{E}f(\bar{\mathbf{x}}(t)) - f(\mathbf{x}^*)). \tag{15}$$

Now we handle the third term in the RHS of Eq. (10) as follows:

$$\mathbb{E}\langle\bar{\nabla}(t), \bar{\mathbf{x}}(t) - \mathbf{x}^*\rangle$$

$$= \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left[\left\langle\nabla f_i(\mathbf{x}_i(t)), \bar{\mathbf{x}}(t) - \mathbf{x}_i(t)\right\rangle + \left\langle\nabla f_i(\mathbf{x}_i(t)), \mathbf{x}_i(t) - \mathbf{x}^*\right\rangle\right]$$

$$\ge \mathbb{E}[f(\bar{\mathbf{x}}(t)) - f(\mathbf{x}^*)] - \frac{L}{2n}\mathbb{E}\|X(t) - \bar{\mathbf{x}}(t)\mathbf{1}^{\mathrm{T}}\|^2. \tag{16}$$

In the first equation, we use $\mathbb{E}_{\zeta_{i,t}}\nabla F_i(\mathbf{x}_i(t),\zeta_{i,t}) = \nabla f_i(\mathbf{x}_i(t))$ for $i \in \mathcal{N}$. The inequality comes from Assumption 4.3. Now, by substituting Eqs. (15) and (16) into Eq. (10), we obtain

$$\mathbb{E}\|\bar{\mathbf{x}}(t+1) - \mathbf{x}^*\|^2$$

$$\le \mathbb{E}\|\bar{\mathbf{x}}(t) - \mathbf{x}^*\|^2 - (2\alpha - 8L\alpha^2)\,[\mathbb{E}f(\bar{\mathbf{x}}(t)) - f(\mathbf{x}^*)] + \frac{\alpha L + 4\alpha^2L^2}{n}\mathbb{E}\|X(t) - \bar{\mathbf{x}}(t)\mathbf{1}^{\mathrm{T}}\|^2 + \frac{2\sigma^2\alpha^2}{n}. \tag{17}$$

Arranging this inequality and choosing $\alpha \le \frac{1}{8L}$, so that $2 - 8L\alpha \ge 1$, there exists

$$\alpha\,[\mathbb{E}f(\bar{\mathbf{x}}(t)) - f(\mathbf{x}^*)] \le \mathbb{E}\|\bar{\mathbf{x}}(t) - \mathbf{x}^*\|^2 - \mathbb{E}\|\bar{\mathbf{x}}(t+1) - \mathbf{x}^*\|^2 + \frac{2\sigma^2\alpha^2}{n} + \frac{3\alpha L}{2n}\mathbb{E}\|X(t) - \bar{\mathbf{x}}(t)\mathbf{1}^{\mathrm{T}}\|^2. \tag{18}$$

By summing both sides for t from 1 to T, and dividing both sides by $T\alpha$, we can find that

$$\frac{1}{T}\sum_{t=1}^{T}(\mathbb{E}f(\bar{\mathbf{x}}(t)) - f(\mathbf{x}^*)) \le \frac{\mathbb{E}\|\mathbf{x}^*\|^2}{T\alpha} + \frac{3L}{2nT}\sum_{t=1}^{T}\mathbb{E}\|X(t) - \bar{\mathbf{x}}(t)\mathbf{1}^{\mathrm{T}}\|^2 + \frac{2\sigma^2\alpha}{n}, \tag{19}$$

where we apply $\bar{\mathbf{x}}(1) = \mathbf{0}$. Next we process $\mathbb{E}\|X(t) - \bar{\mathbf{x}}(t)\mathbf{1}^{\mathrm{T}}\|^2$. Applying updating rules (8) and the recursion technique, we have

$$
\begin{aligned}
&X(t)\\
&= X(t-1)A + \big[\hat{X}(t) - X(t-1)\big](A-\mathbf{I}) - \alpha\nabla(t-1)\\
&= \Big\{X(t-2)A + \big[\hat{X}(t-1) - X(t-2)\big](A-\mathbf{I}) - \alpha\nabla(t-2)\Big\}A\\
&\quad + \big[\hat{X}(t) - X(t-1)\big](A-\mathbf{I}) - \alpha\nabla(t-1)\\
&= X(1)A^{t-1} + \sum_{k=0}^{t-2}\big[\hat{X}(t-k) - X(t-1-k)\big](A-\mathbf{I})A^k - \alpha\sum_{k=0}^{t-2}\nabla(t-1-k)A^k. \qquad (20)
\end{aligned}
$$

Since A is doubly stochastic, we get

$$
\bar{\mathbf{x}}(t)\mathbf{1}^{\mathrm{T}} = X(t)\frac{\mathbf{1}\mathbf{1}^{\mathrm{T}}}{n} = X(1)\frac{\mathbf{1}\mathbf{1}^{\mathrm{T}}}{n} - \alpha\sum_{k=0}^{t-2}\nabla(t-1-k)\frac{\mathbf{1}\mathbf{1}^{\mathrm{T}}}{n}. \qquad (21)
$$

Therefore, after t iterations, the difference between the current parameter and the global average can be expressed as

$$
\begin{aligned}
&\mathbb{E}\|X(t) - \bar{\mathbf{x}}(t)\mathbf{1}^{\mathrm{T}}\|^2\\
&= \mathbb{E}\left\|\sum_{k=0}^{t-2}\big[\hat{X}(t-k) - X(t-1-k)\big](A-\mathbf{I})A^k - \alpha\sum_{k=0}^{t-2}\nabla(t-1-k)(A^k - \frac{\mathbf{1}\mathbf{1}^{\mathrm{T}}}{n})\right\|^2\\
&\le 2\mathbb{E}\left\|\sum_{k=0}^{t-2}\big[\hat{X}(t-k) - X(t-1-k)\big](A-\mathbf{I})A^k\right\|^2 + 2\alpha^2\mathbb{E}\left\|\sum_{k=0}^{t-2}\nabla(t-1-k)(A^k - \frac{\mathbf{1}\mathbf{1}^{\mathrm{T}}}{n})\right\|^2 \qquad (22)
\end{aligned}
$$

where we use $X(1) = 0$ in the equality. With Lemma 4.1 and $\|A-\mathbf{I}\| \le \|A\| + \|\mathbf{I}\| \le 2\sqrt{n}$, we can bound these two terms as follows:

$$
\begin{aligned}
&2\mathbb{E}\left\|\sum_{k=0}^{t-2}\big[\hat{X}(t-k) - X(t-1-k)\big](A-\mathbf{I})A^k\right\|^2\\
&= 2\mathbb{E}\left\|\sum_{k=0}^{t-2}\big[\hat{X}(t-k) - X(t-1-k)\big](A-\mathbf{I})(A^k - \frac{\mathbf{1}\mathbf{1}^{\mathrm{T}}}{n})\right\|^2\\
&\le 2\mathbb{E}\left(\sum_{k=0}^{t-2}\left\|\big[\hat{X}(t-k) - X(t-1-k)\big](A-\mathbf{I})(A^k - \frac{\mathbf{1}\mathbf{1}^{\mathrm{T}}}{n})\right\|\right)^2\\
&\le 8nC^2\mathbb{E}\left(\sum_{k=0}^{t-2}\lambda^k\left\|\hat{X}(t-k) - X(t-1-k)\right\|\right)^2, \qquad (23)
\end{aligned}
$$

and

$$
\begin{aligned}
&2\alpha^2\mathbb{E}\left\|\sum_{k=0}^{t-2}\nabla(t-1-k)(A^k-\frac{\mathbf{1}\mathbf{1}^{\mathrm{T}}}{n})\right\|^2\\
&\leq 2\alpha^2\mathbb{E}\left(\sum_{k=0}^{t-2}\left\|\nabla(t-1-k)(A^k-\frac{\mathbf{1}\mathbf{1}^{\mathrm{T}}}{n})\right\|\right)^2\\
&\leq 2\alpha^2C^2\mathbb{E}\left(\sum_{k=0}^{t-2}\lambda^k\left\|\nabla(t-1-k)\right\|\right)^2
\end{aligned}
\tag{24}
$$

Now, substituting Eqs. (23) and (24) to Eq. (22), we have

$$
\begin{aligned}
&\mathbb{E}\left\|X(t)-\bar{\mathbf{x}}(1)\mathbf{1}^{\mathrm{T}}\right\|^2\\
&\leq 8nC^2\mathbb{E}\left(\sum_{k=0}^{t-2}\lambda^k\left\|\hat{X}(t-k)-X(t-1-k)\right\|\right)^2+2\alpha^2C^2\mathbb{E}\left(\sum_{k=0}^{t-2}\lambda^k\left\|\nabla(t-1-k)\right\|\right)^2.
\end{aligned}
\tag{25}
$$

We process the first term of RHS of Eq. (25) in the following way:

$$
\begin{aligned}
&\left(\sum_{k=0}^{t-2}\lambda^k\left\|\hat{X}(t-k)-X(t-1-k)\right\|\right)^2=\sum_{k=0}^{t-2}\lambda^{2k}\left\|\hat{X}(t-k)-X(t-1-k)\right\|^2\\
&\quad+\sum_{k\neq k'}\lambda^k\lambda^{k'}\left\|\hat{X}(t-k)-X(t-1-k)\right\|\cdot\left\|\hat{X}(t-k')-X(t-1-k')\right\|\\
&\quad\leq\sum_{k=0}^{t-2}\lambda^{2k}\left\|\hat{X}(t-k)-X(t-1-k)\right\|^2+\frac{1}{2}\sum_{k\neq k'}\lambda^k\lambda^{k'}\left\|\hat{X}(t-k)-X(t-1-k)\right\|^2\\
&\quad+\frac{1}{2}\sum_{k\neq k'}\lambda^k\lambda^{k'}\left\|\hat{X}(t-k')-X(t-1-k')\right\|^2\\
&\quad\leq\sum_{k=0}^{t-2}\left(\lambda^{2k}+\frac{\lambda^k}{1-\lambda}\right)\left\|\hat{X}(t-k)-X(t-1-k)\right\|^2\\
&\quad\leq\sum_{k=0}^{t-2}\frac{2\lambda^k}{1-\lambda}\left\|\hat{X}(t-k)-X(t-1-k)\right\|^2,
\end{aligned}
\tag{26}
$$

where the last inequality comes from $\lambda^k\leq\dfrac{1}{1-\lambda}$. Applying the same technique to the second term of Eq. (25), we can get

$$
\mathbb{E}\left(\sum_{k=0}^{t-2}\lambda^k\mathbb{E}\left\|\nabla(t-1-k)\right\|\right)^2\leq\sum_{k=0}^{t-2}\frac{2\lambda^k}{1-\lambda}\mathbb{E}\left\|\nabla(t-1-k)\right\|^2\leq\frac{2nM^2}{(1-\lambda)^2},
\tag{27}
$$

where the last inequality applies Assumption 4.5. Then Eq. (25) can be written as

$$\mathbb{E}\left\|X(t)-\bar{\mathbf{x}}(t)\mathbf{1}^{\mathrm{T}}\right\|^2 \leq 16nC^2\sum_{k=0}^{t-2}\frac{\lambda^k}{1-\lambda}\mathbb{E}\left\|\hat{X}(t-k)-X(t-1-k)\right\|^2+\frac{4C^2\alpha^2nM^2}{(1-\lambda)^2}. \tag{28}$$

This motivates us to bound $\left\|\hat{X}(t+2)-X(t+1)\right\|^2$ as follows:

$$\begin{aligned}
\mathbb{E}\left\|\hat{X}(t+2)-X(t+1)\right\|^2 &= \mathbb{E}\left\|\hat{X}(t+1)+Q(t+1)-X(t+1)\right\|^2\\
&= \mathbb{E}\left\|Q\big(X(t+1)-\hat{X}(t+1)\big)-\big(X(t+1)-\hat{X}(t+1)\big)\right\|^2\\
&\leq (1-\omega)\mathbb{E}\left\|X(t+1)-\hat{X}(t+1)\right\|^2\\
&\leq 3(1-\omega)\Big(\mathbb{E}\left\|X(t)-\hat{X}(t+1)\right\|^2+\alpha^2\mathbb{E}\left\|\nabla(t)\right\|^2+\mathbb{E}\left\|\hat{X}(t+1)(A-\mathbf{I})\right\|^2\Big)\\
&\leq 3(1-\omega)\mathbb{E}\left\|X(t)-\hat{X}(t+1)\right\|^2+3(1-\omega)\alpha^2nM^2\\
&\quad+6(1-\omega)\mathbb{E}\left\|(\hat{X}(t+1)-X(t))(A-\mathbf{I})\right\|^2\\
&\quad+6(1-\omega)\mathbb{E}\left\|\big(X(t)-\bar{\mathbf{x}}(t)\mathbf{1}^{\mathrm{T}}\big)(A-\mathbf{I})\right\|^2\\
&\leq (3+24n)(1-\omega)\mathbb{E}\left\|X(t)-\hat{X}(t+1)\right\|^2\\
&\quad+3(1-\omega)\alpha^2nM^2+24n(1-\omega)\mathbb{E}\left\|X(t)-\bar{\mathbf{x}}(t)\mathbf{1}^{\mathrm{T}}\right\|^2,
\end{aligned} \tag{29}$$

where the second and the last inequalities come from Eq. (8) and $\|A-\mathbf{I}\| \leq 2\sqrt{n}$, respectively. For simplicity, we use $L(t)$ to denote $\mathbb{E}\|\hat{X}(t+1)-X(t)\|^2$ and use $K(t)$ to denote $\mathbb{E}\|X(t)-\bar{\mathbf{x}}(t)\mathbf{1}^{\mathrm{T}}\|^2$. Then Eqs. (28) and (29) can be presented as

$$\begin{aligned}
&L(t+1) \leq (3+24n)(1-\omega)L(t)+3(1-\omega)\alpha^2nM^2+24n(1-\omega)K(t),\\
&K(t) \leq 16nC^2\sum_{k=0}^{t-2}\frac{\lambda^k}{1-\lambda}L(t-1-k)+\frac{4C^2\alpha^2nM^2}{(1-\lambda)^2}.
\end{aligned} \tag{30}$$

Next we use induction to prove that when $\omega > 1-\dfrac{1}{3+24n+\dfrac{384n^2C^2}{(1-\lambda)^2}}$, we have $L(t) < \beta\alpha^2$, for some $\beta > 0$.

When $t=1$, we have $L(1)=\mathbb{E}\|\hat{X}(2)-X(1)\|^2=0$. Now we assume that the proposition is true for any index less than or equal to t. Then there exists

$$\begin{aligned}
L(t+1) &< (3+24n)(1-\omega)\beta\alpha^2+3(1-\omega)\alpha^2nM^2+\frac{96(1-\omega)C^2n^2M^2}{(1-\lambda)^2}\alpha^2+\frac{384n^2(1-\omega)C^2}{(1-\lambda)^2}\beta\alpha^2\\
&= \left[(3+24n)(1-\omega)+\frac{384n^2(1-\omega)C^2}{(1-\lambda)^2}\right]\beta\alpha^2+\left[3(1-\omega)nM^2+\frac{96(1-\omega)C^2n^2M^2}{(1-\lambda)^2}\right]\alpha^2.
\end{aligned} \tag{31}$$

By setting $\omega > 1 - \dfrac{1}{3 + 24n + \dfrac{384n^2C^2}{(1-\lambda)^2}}$ and choosing

$$\beta = \frac{3(1-\omega)nM^2 + \dfrac{96(1-\omega)C^2n^2M^2}{(1-\lambda)^2}}{1 - [(3+24n)(1-\omega) + \dfrac{384n^2(1-\omega)C^2}{(1-\lambda)^2}]},$$

the RHS of Eq. (31) will be less than $\beta\alpha^2$, and the proposition is proved.

With $L(t) < \beta\alpha^2$, we have

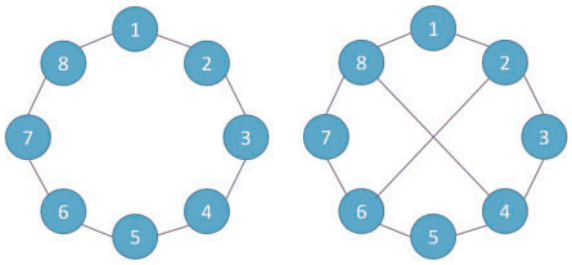
$$\mathbb{E}\|X(t) - \bar{\mathbf{x}}(t)\mathbf{1}^{\mathrm{T}}\|^2 = K(t) \leq \frac{16nC^2\beta\alpha^2}{(1-\lambda)^2} + \frac{4C^2\alpha^2nM^2}{(1-\lambda)^2}. \tag{32}$$

By substituting this result to Eq. (19), we can get Theorem 4.1.

5 Experiments

5.1 Evaluation Setup

Testbed setting and topology. Our experiments are conducted on a testbed of 8 servers, each with an 8-core Intel Xeon W2140b CPU at 3.20 GHz, a 32 GB DDR4 RAM and a Mellanox Connect-X3 NIC supporting 10 GbE links. We use PyTorch as the experiment platform and employ its Distributed Data-Parallel Training (DDP) paradigm to realize parameter communication among the 8 servers. The main topology in experiments is a ring topology, as shown in Fig. 1a. Ring topology can be used in Local Area Network (LAN) or Wide Area Network (WAN) networks, and is also commonly used [28] in distributed learning.



(a) bipartite topology **(b)** sparse topology

Figure 1: Undirected topologies with 8 nodes

We consider a task where all nodes aim to classify images in the MNIST or CIFAR-10 dataset into 10 classes using a linear neural network. The key features of both datasets are listed in Table 2. The objective function for this task is cross-entropy loss. The cross-entropy loss of a data batch is defined as

$$\frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{l} p(x_{ij})\log(q(x_{ij})),$$

where m and l represent the batch size and the number of categories, respectively. In this equation, $q(x_{ij})$ denotes the predicted probability of sample i belonging to class j, while $p(x_{ij})$ represents the true

label, which is equal to 1 when sample i belongs to class j and 0 otherwise. The entire training dataset is divided into 8 equal partitions, each stored in an independent node. Unless otherwise specified, the learning rate and batch size are set to be 0.01 and 64, respectively.

Table 2: Key features of datasets

Datasets	Train size	Feature dimension	Number of classes
MNIST	60,000	784	10
CIFAR-10	50,000	1,024	10

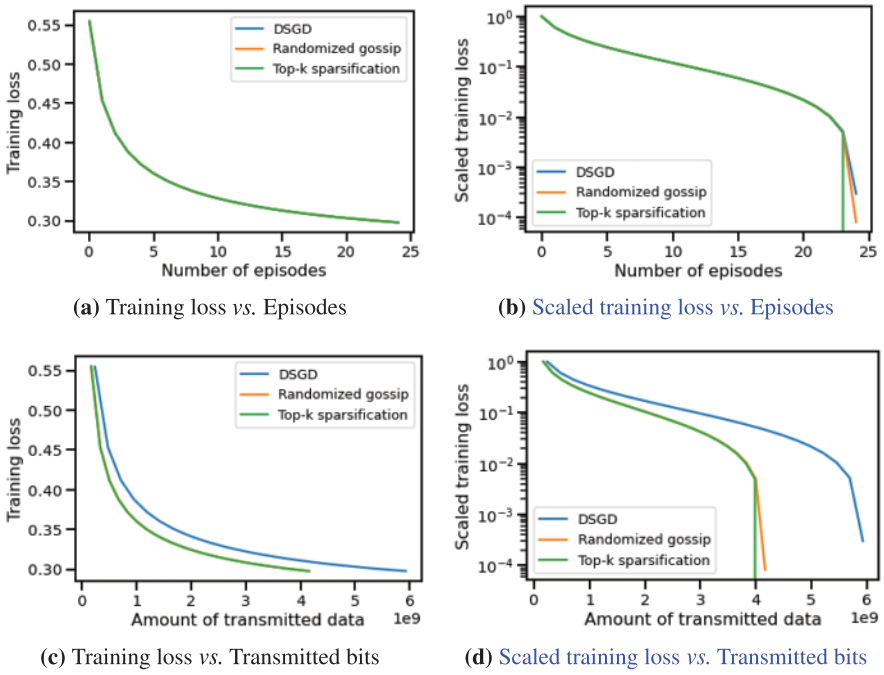
5.2 Basic Results

We evaluate the performance of the CHOCO-SGD algorithm in comparison to the DSGD baseline. In CHOCO-SGD, we examine the efficacy of two compression operators outlined in Example 1: randomized gossip with $p = 0.7$ and Top-k sparsification with $\omega = 0.7$. To align with our theoretical convergence criterion, the performance is evaluated based on the global loss computed on the global average parameter, denoted as $f(\bar{\mathbf{x}}(t))$. The model is trained for 25 episodes, where each episode corresponds to a pass through the entire dataset.

The results in Fig. 2a demonstrate that CHOCO-SGD and DSGD exhibit comparable convergence rates, with similar reductions in loss after the same number of episodes. To distinguish the curves, we scale the loss values to the range [0, 1] with the following formula:

$$v_{scale} = \frac{v - v_{min}}{v_{max} - v_{min}},$$

where v is the real loss value in the training process, v_{min} and v_{max} denote the minimal and the maximal loss value achieved by three algorithms, respectively, and v_{scale} represents the scaled loss value. The scaled results are depicted in Fig. 2b on a logarithmic scale. In the following context, the scaled training loss is processed likewise. In Fig. 2b, CHOCO-SGD and DSGD show only marginal differences in convergence rate. Furthermore, Figs. 2c and 2d (scaled) illustrate the comparison of the loss with respect to the number of transmitted bits. Remarkably, CHOCO-SGD, employing either randomized gossip or Top-k sparsification methods, achieves lower loss values while transmitting the same volume of data. This outcome can be attributed to the compression of certain values, which enables nodes to transmit fewer data in each communication round without adversely affecting the convergence rate. Similar results also appear on CIFAR-10 datasets, as shown in Fig. 3. CHOCO-SGD transmits about 1.2×10^{10} bits, while DSGD needs to transmit 1.7×10^{10} bits, when they achieve the same training error. These results align with our theoretical analysis, indicating that compression can effectively reduce the amount of transmitted data while exerting minimal influence on the convergence rate for non-strongly convex objectives, provided that the fidelity threshold is appropriately maintained.



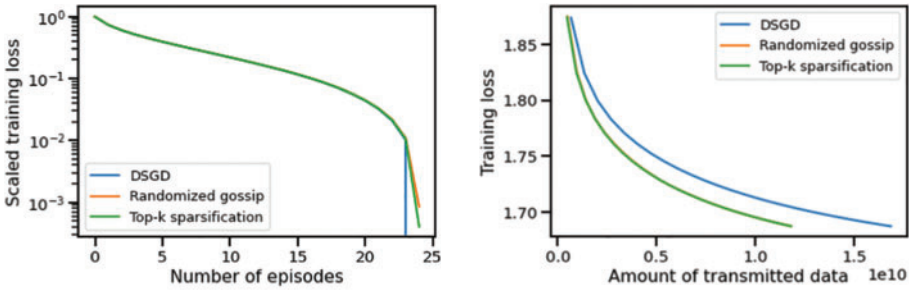
(a) Training loss *vs.* Episodes

(b) Scaled training loss *vs.* Episodes

(c) Training loss *vs.* Transmitted bits

(d) Scaled training loss *vs.* Transmitted bits

Figure 2: Training loss *vs.* episodes/transmitted data amount of CHOCO-SGD and DSGD on MNIST Dataset



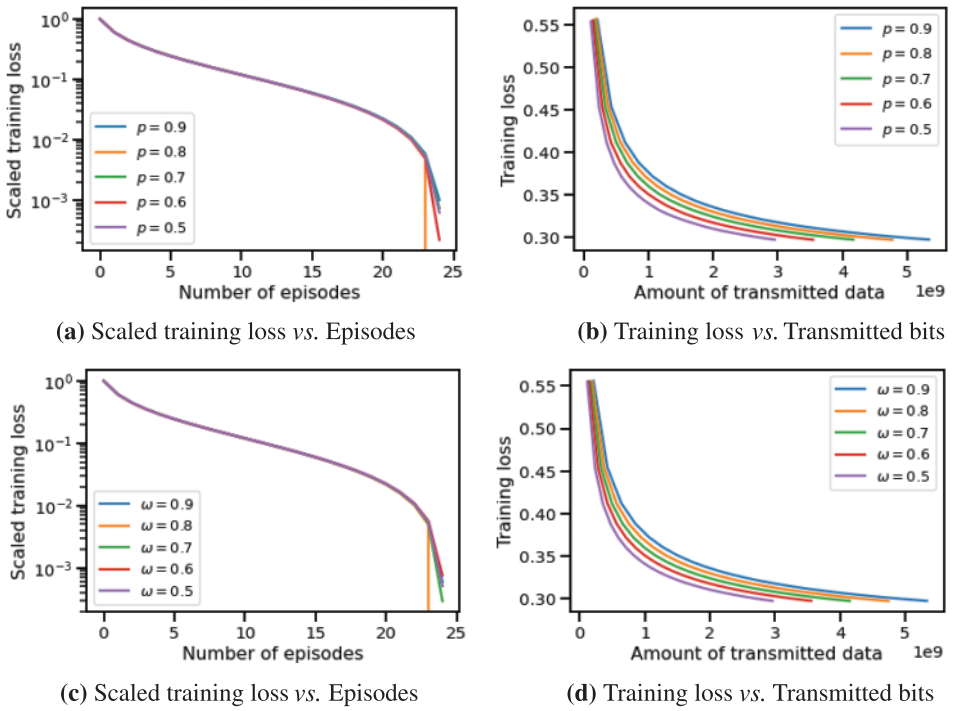
(a) Scaled training loss *vs.* Episodes

(b) Training loss *vs.* Transmitted bits

Figure 3: Training loss *vs.* episodes/transmitted data amount of CHOCO-SGD and DSGD on CIFAR-10 Dataset

5.3 Sensitivity to Compression Fidelity

To further investigate the impact of compression fidelity, we conduct several experiments by controlling the values of p and ω within the range of 0.9 to 0.5 for two compression operators. In randomized gossip, a smaller value of p increases the probability of transmitting full-zero parameters, which could lead to inaccuracies in $\hat{\mathbf{x}}_j$ as described in Algorithm 2. The change in (scaled) training loss with respect to iterations and transmitted bits is depicted in Figs. 4a, 4b (MNIST) and Figs. 5a, 5b (CIFAR-10), respectively. We observe that lowering the value of p results in fewer transmission bits without slowing down the convergence rate. Similarly, the results obtained using Top-k sparsification, as shown in Figs. 4c, 4d and 5c, 5d indicate that reducing the number of bits, while maintaining ω within a certain threshold, ensures satisfactory convergence rates and reduces communication burden.



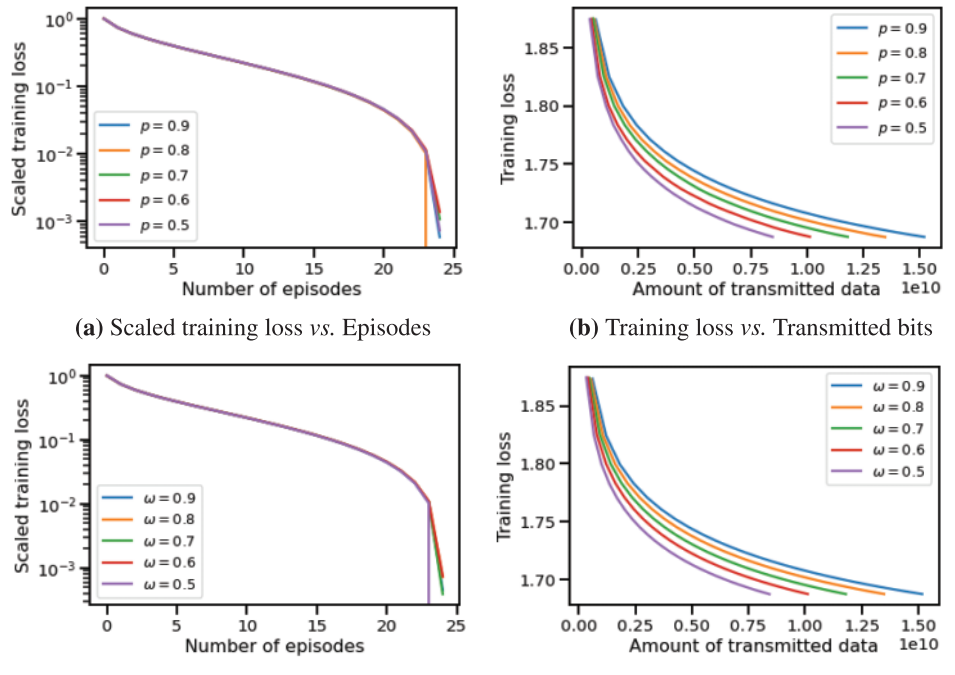
(a) Scaled training loss *vs.* Episodes

(b) Training loss *vs.* Transmitted bits

(c) Scaled training loss *vs.* Episodes

(d) Training loss *vs.* Transmitted bits

Figure 4: Training loss of CHOCO-SGD with randomized gossip (a), (b) and with Top-k sparsification (c), (d) on MNIST dataset. Both p and ω vary from 0.9 to 0.5



(a) Scaled training loss *vs.* Episodes

(b) Training loss *vs.* Transmitted bits

(c) Scaled training loss *vs.* Episodes

(d) Training loss *vs.* Transmitted bits

Figure 5: Training loss of CHOCO-SGD with randomized gossip (a), (b) and with Top-k sparsification (c), (d) on CIFAR-10 dataset. Both p and ω vary from 0.9 to 0.5

Furthermore, we conduct experiments to determine if CHOCO-SGD fails to converge when the compression fidelity is small for non-strongly convex objectives, as stated in Eq. (6). Specifically, for randomized gossip and Top-k sparsification, we set both p and ω to 0.4. Fig. 6 demonstrates that in an over-compression scenario, the training loss of CHOCO-SGD continues to increase, implying that the algorithm fails to converge. With these results, we empirically verify that within the context of non-strongly convex functions, CHOCO-SGD requires control of the compression fidelity within a threshold to ensure convergence, which differs from the situation with strongly convex functions. In other words, CHOCO-SGD cannot achieve arbitrary compression in a non-strongly convex scenario.

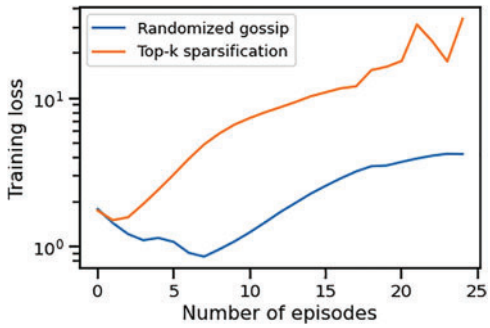


Figure 6: Convergence of CHOCO-SGD with low compression fidelity on MNIST dataset

5.4 Robustness towards Learning Hyperparameters

In this section, we study the influence of hyperparameters (learning rate α and batch size b), on the convergence performance of the CHOCO-SGD algorithm for non-strongly convex functions. The compression fidelity is set to be $p = \omega = 0.7$. We test on three learning rates $\alpha = 0.01, 0.05, 0.1$ and three batch sizes $b = 32, 64, 128$. The results are displayed in Tables 3 and 4. With the increase of the batch size, the final loss values of three algorithms after 25 training episodes become larger. Because each episode is one pass through the whole dataset, and larger batch size leads to fewer batches and thus fewer times of gradient descent and data transmission. Furthermore, the loss goes smaller when the learning rate gets larger, since the global parameter takes a larger step towards the global minimum. No matter in which hyperparameter setting, CHOCO-SGD and DSGD algorithms can achieve similar convergence error with the same training episodes, but CHOCO-SGD transmits less data amount. Furthermore, according to Table 4, the change of the learning rate has negligible influence on communication load, while larger batch size leads to fewer transmission data.

Table 3: Summary of training loss with different hyperparameter settings. The loss values are recorded after 25 training episodes and are rounded to three decimal places

	$\alpha = 0.01$			$\alpha = 0.05$			$\alpha = 0.1$		
	DSGD	Randomized gossip	Top-k sparsification	DSGD	Randomized gossip	Top-k sparsification	DSGD	Randomized gossip	Top-k sparsification
b = 32	2.793	2.794	2.793	2.534	2.530	2.535	2.578	2.598	2.565
b = 64	2.946	2.976	2.975	2.628	2.628	2.628	2.549	2.568	2.557
b = 128	3.223	3.224	3.224	2.743	2.744	2.743	2.620	2.620	2.620

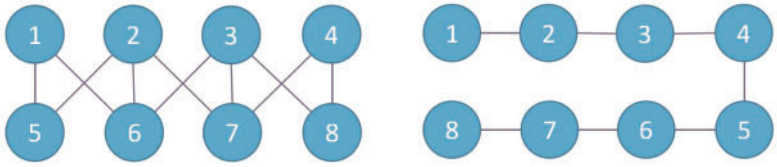
Table 4: Summary of Transmitted bits with different hyperparameter settings. The bits values are recorded after 25 training episodes and are rounded to three decimal places

	$\alpha = 0.01$			$\alpha = 0.05$			$\alpha = 0.1$		
	DSGD	Randomized gossip	Top-k sparsification	DSGD	Randomized gossip	Top-k sparsification	DSGD	Randomized gossip	Top-k sparsification
b = 32	1.033	0.724	0.723	1.033	0.723	0.723	1.033	0.723	0.723
b = 64	0.593	0.416	0.415	0.518	0.364	0.363	0.519	0.364	0.363
b = 128	0.259	0.183	0.182	0.259	0.181	0.182	0.259	0.182	0.182

5.5 Applicability to Various Topologies

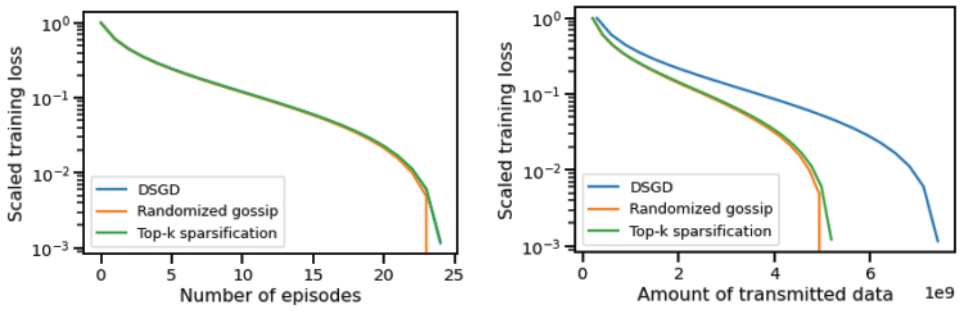
5.5.1 Investigating Different Fixed Topologies

To explore the algorithm's applicability for non-strongly convex objectives, we test the convergence performance of CHOCO-SGD on two different topologies, as displayed in Fig. 7. Fig. 7a shows a bipartite topology, where nodes $1-4$ and $5-8$ are divided into two groups and each edge can only connect nodes in different groups. Besides, Fig. 7b displays a more sparse topology, where parameters can only be transmitted along one path in two directions. The compression fidelity is set to be $p = \omega = 0.7$ which is the same as basic experiments. Figs. 8 and 9 show the scaled training loss of MNIST dataset training on two topologies, respectively. In Fig. 8, CHOCO-SGD and DSGD have the same convergence speed, since they have similar loss values after the same training episodes. But the transmission amount of CHOCO-SGD, about 5×10^9, is much less than 7×10^9 of DSGD. Besides, in the sparser topology, Fig. 9b demonstrates that both algorithms can transmit fewer data to achieve convergence, but the transmitted amount of DSGD is 40% more than that of CHOCO-SGD. Therefore, these results further illustrate the robustness and applicability of CHOCO-SGD for non-strongly convex functions on various topologies.



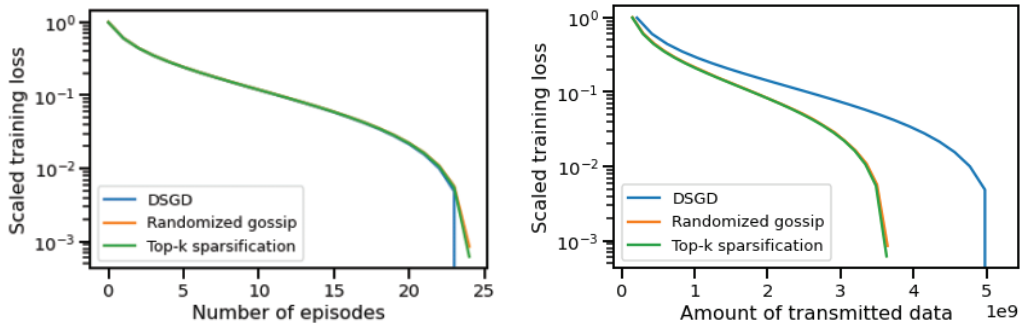
(a) bipartite topology **(b)** sparse topology

Figure 7: Different undirected and connected topologies



(a) Scaled training loss *vs.* Episodes **(b)** Scaled training loss *vs.* Transmitted bits

Figure 8: Training loss *vs.* episodes/transmitted data amount of CHOCO-SGD and DSGD on bipartite topology Fig. 7a



(a) Scaled training loss *vs.* Episodes **(b)** Scaled training loss *vs.* Transmitted bits

Figure 9: Training loss *vs.* episodes/transmitted data amount of CHOCO-SGD and DSGD on sparse topology Fig. 7b

5.5.2 Time-Varying Topologies

We further explore the impact of time-varying topologies on the convergence of CHOCO-SGD. The training topology alternates between the two graphs in Fig. 1. For instance, if the nodes are initially distributed over the left topology, in the next iteration, they will be connected according to the right topology. It is important to note that this setup still ensures connectivity (as stated in Assumption 4.1). We set the compression coefficients to $p = \omega = 0.9$ to reduce their influence on the convergence. Fig. 10 illustrates that CHOCO-SGD is not able to converge in a non-strongly convex and time-varying environment. Because some nodes receive parameters from specific neighbors inconsistently, resulting in an uncorrectable bias from true parameters and the algorithm diverges eventually.

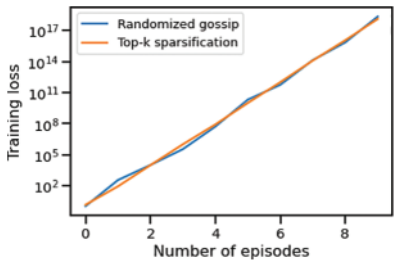


Figure 10: Convergence of CHOCO-SGD on time-varying topologies

6 Conclusion and Future Work

In this work, we study CHOCO-SGD algorithm in the case that the objective function is non-strongly convex. We provide the first theoretical analysis of this situation and prove that when controlling compression fidelity within a certain threshold, it has the same convergence rate order as DSGD. Experimental results validate the theoretical analysis by demonstrating that CHOCO-SGD converges more quickly than DSGD when transmitting the same data amount. Besides, a low compression fidelity and time-varying topology can make CHOCO-SGD not converge in the end.

There are several open topics worthy of investigating in our future work. The first one is to explore CHOCO-SGD's performance in more complex non-convex scenarios, which can provide a deeper understanding of its capabilities. Besides, it will be meaningful to improve this algorithm to achieve arbitrary compression fidelity for non-strongly convex functions, since most real-world problems are non-strongly convex and such improvement can enhance CHOCO-SGD's durability and applicability.

Acknowledgement: The authors are grateful for the theoretical support by Simiao Jiao. Besides, this paper's logical rigor, integrity, and content quality have been greatly enhanced, so the authors also wish to express their appreciation to anonymous reviewers and journal editors for assistance.

Funding Statement: This work was supported in part by the Shanghai Natural Science Foundation under the Grant 22ZR1407000.

Author Contributions: The authors' contributions to this manuscript are as follows: study conception: X.L., Y.X.; experiment design and data collection: X.L.; theoretical analysis, interpretation of results, and draft manuscript preparation: X.L., Y.X. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The data, including MNIST and CIFAR-10 datasets, used in this manuscript is accessible via the following link: https://www.kaggle.com/datasets.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Jiang, X., Hu, Z., Wang, S., Zhang, Y. (2023). A survey on artificial intelligence in posture recognition. *Computer Modeling in Engineering & Sciences, 137(1),* 35–82. https://doi.org/10.32604/cmes.2023.027676
2. Koloskova, A., Stich, S., Jaggi, M. (2019). Decentralized stochastic optimization and gossip algorithms with compressed communication. *International Conference on Machine Learning*, Long Beach, California, USA.

3. Allen-Zhu, Z., Yuan, Y. (2016). Improved SVRG for non-strongly-convex or sum-of-non-convex objectives. *International Conference on Machine Learning*, New York, NY, USA.
4. Rajalakshmi, M., Rengaraj, R., Bharadwaj, M., Kumar, A., Naren Raju, N. et al. (2018). An ensemble based hand vein pattern authentication system. *Computer Modeling in Engineering & Sciences, 114(2),* 209–220. https://doi.org/10.3970/cmes.2018.114.209
5. Saber, R. O., Murray, R. M. (2003). Consensus protocols for networks of dynamic agents. *Proceedings of the 2003 American Control Conference*, pp. 951–956. Denver, CO, USA.
6. Xiao, L., Boyd, S. (2004). Fast linear iterations for distributed averaging. *Systems & Control Letters, 53(1),* 65–78.
7. Nedic, A., Ozdaglar, A. (2009). Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control, 54(1),* 48–61.
8. Yuan, K., Ling, Q., Yin, W. (2016). On the convergence of decentralized gradient descent. *SIAM Journal on Optimization, 26(3),* 1835–1854.
9. Sun, H., Lu, S., Hong, M. (2020). Improving the sample and communication complexity for decentralized non-convex optimization: Joint gradient estimation and tracking. *International Conference on Machine Learning*, pp. 9154–9165. International Machine Learning Society (IMLS).
10. Tang, H., Gan, S., Zhang, C., Zhang, T., Liu, J. (2018). Communication compression for decentralized training. In: *Advances in neural information processing systems 31,* pp. 7652–7662.
11. Sun, Y., Scutari, G., Daneshmand, A. (2022). Distributed optimization based on gradient tracking revisited: Enhancing convergence rate via surrogation. *SIAM Journal on Optimization, 32(2),* 354–385.
12. Jiang, X., Zeng, X., Sun, J., Chen, J. (2023). Distributed proximal gradient algorithm for non-convex optimization over time-varying networks. *IEEE Transactions on Control of Network Systems, 10(2),* 1005–1017.
13. Lian, X., Zhang, C., Zhang, H., Hsieh, C. J., Zhang, W. et al. (2017). Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent. In: *Advances in neural information processing systems*. Long Beach, California, USA.
14. Shi, W., Ling, Q., Wu, G., Yin, W. (2015). Extra: An exact first-order algorithm for decentralized consensus optimization. *SIAM Journal on Optimization, 25(2),* 944–966.
15. Assran, M., Loizou, N., Ballas, N., Rabbat, M. (2019). Stochastic gradient push for distributed deep learning. *International Conference on Machine Learning*, Long Beach, California, USA, PMLR.
16. Nedić, A., Olshevsky, A., Rabbat, M. G. (2018). Network topology and communication-computation tradeoffs in decentralized optimization. *Proceedings of the IEEE, 106(5),* 953–976.
17. Nedic, A. (2020). Distributed gradient methods for convex machine learning problems in networks: Distributed optimization. *IEEE Signal Processing Magazine, 37(3),* 92–101.
18. Li, Z., Richtárik, P. (2021). Canita: Faster rates for distributed convex optimization with communication compression. In: *Advances in neural information processing systems 34,* pp. 13770–13781.
19. Tang, Z., Shi, S., Li, B., Chu, X. (2022). GossipFL: A decentralized federated learning framework with sparsified and adaptive communication. *IEEE Transactions on Parallel and Distributed Systems, 34(3),* 909–922.
20. Stich, S. U., Cordonnier, J. B., Jaggi, M. (2018). Sparsified sgd with memory. In: *Advances in neural information processing systems*. Montréal, Canada.
21. Wangni, J., Wang, J., Liu, J., Zhang, T. (2018). Gradient sparsification for communication-efficient distributed optimization. In: *Advances in neural information processing systems 31,* pp. 1299–1309.
22. Bernstein, J., Wang, Y. X., Azizzadenesheli, K., Anandkumar, A. (2018). signSGD: Compressed optimisation for non-convex problems. *International Conference on Machine Learning*, Stockholm, Sweden.
23. Reisizadeh, A., Mokhtari, A., Hassani, H., Pedarsani, R. (2019). An exact quantized decentralized gradient descent algorithm. *IEEE Transactions on Signal Processing, 67(19),* 4934–4947.

24. Karimireddy, S. P., Rebjock, Q., Stich, S., Jaggi, M. (2019). Error feedback fixes signsgd and other gradient compression schemes. *International Conference on Machine Learning*, Long Beach, California, USA.
25. Taheri, H., Mokhtari, A., Hassani, H., Pedarsani, R. (2020). Quantized decentralized stochastic learning over directed graphs. *International Conference on Machine Learning*, vol. 119, pp. 9324–9333.
26. DeGroot, M. H. (1974). Reaching a consensus. *Journal of the American Statistical Association, 69(345),* 118–121.
27. Merris, R. (1997). Doubly stochastic graph matrices. *Publikacije Elektrotehničkog fakulteta. Serija Matematika, (8),* 64–71.
28. Wang, Z., Hu, Y., Yan, S., Wang, Z., Hou, R. et al. (2022). Efficient ring-topology decentralized federated learning with deep generative models for medical data in ehealthcare systems. *Electronics, 11(10),* 1548.

Appendix A. Proof of Lemma 4.1

We firstly prove that for any matrix $M \in \mathbb{R}^{d\times n}$, if $M\mathbf{1} = 0$, we have $\|MA\| \leq \sigma_2\|M\|$. Assume that with singular value decomposition, $A = U\Sigma V^{\mathrm{T}}$, where U is a matrix consists of eigenvectors of AA^{T}. If $U = (u_1, \cdots, u_n)$, because AA^{T} has eigenvalue 1, we can choose u_1 to be $\mathbf{1}$, which is the eigenvector of eigenvalue 1. Then

$$\|MA\| = \|MU\Sigma V^{\mathrm{T}}\| = \|MU\Sigma\| = \left\|(M\mathbf{1}, u_2, \cdots, u_n) \cdot diag\{1, \sigma_2, \cdots\}\right\| \leq \|M\|, \tag{33}$$

and

$$\left\|A^t - \frac{1}{n}\mathbf{1}\mathbf{1}^{\mathrm{T}}\right\| = \left\|[A^{t-1} - \frac{1}{n}\mathbf{1}\mathbf{1}^{\mathrm{T}}]A\right\| \leq \sigma_2. \tag{34}$$

We can obtain

$$\left\|A^t - \frac{1}{n}\mathbf{1}\mathbf{1}^{\mathrm{T}}\right\| \leq \sigma_2^t\left\|\mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^{\mathrm{T}}\right\| \leq C\sigma_2^t, \tag{35}$$

where $C = \|\mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^{\mathrm{T}}\| = \sqrt{n-1}$.